6-2020

RIT-Eyes: Realistic Eye Image and Video Generation for Eye Tracking Applications

Nitinraj Nair
nrn2741@rit.edu

RIT-Eyes: Realistic Eye Image and Video Generation for Eye Tracking Applications

by

Nitinraj Nair

A Thesis Submitted
in
Partial Fulfillment of the
Requirements for the Degree of
Master of Science
in
Computer Science

Supervised by

Dr. Reynold Bailey
Professor
Department of Computer Science
Rochester Institute of Technology

Department of Computer Science

B. Thomas Golisano College of Computing and Information Sciences

Rochester Institute of Technology
Rochester, New York

June 2020

Acknowledgments

I wish to express my very great appreciation to Dr. Reynold Bailey for advising me and providing me an opportunity to work on this project under his guidance.

I would like to offer special thanks to my collaborators Rakshit Khothari, Aayush Chaudaury and Zhizhuo Yang for the many discussions and technical assistance that helped to advance this project. I also wish to express my gratitude to Dr. Gabriel Diaz and Dr. Jeff Plez who helped me by providing the guidance and their expert advice during the period of my project work.

I would like to thank Dr. Erroll Wood and Dr. John Daugman for providing infrared and RGB iris textures which were used in the rendering pipeline.

Abstract

RIT-Eyes: Realistic Eye Image and Video Generation for Eye Tracking Applications

Nitinraj Nair

Supervising Professor: Dr. Reynold Bailey
Professor
Department of Computer Science
Rochester Institute of Technology

Deep neural networks for video-based eye tracking have demonstrated resilience to noisy environments, stray reflections, and low resolution. However, to train these networks, a large number of manually annotated images are required. To alleviate the cumbersome process of manual labeling, computer graphics rendering is employed to automatically generate a large corpus of annotated eye images under various conditions. In this work, we introduce a synthetic eye image and video generation platform called RIT-Eyes that improves upon previous work by adding features such as an active deformable iris, an aspherical cornea, retinal retro-reflection, and gaze-coordinated eye-lid deformations. To demonstrate the utility of our platform, we render images reflecting the represented gaze distributions inherent in two publicly available eye image datasets, NVGaze and OpenEDS. Additionally, we also render two datasets which mimic the characteristics of Pupil Labs Core mobile eye tracker. Our platform enables users to render realistic eye images by providing parameters for camera position, illuminator position, and head and eye pose. The

pipeline can also be used to render temporal sequences of realistic eye movements captured in datasets such as Gaze-in-Wild.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Gaze estimation is very important in order to understand human visual perception. There are many applications for this which includes human machine interaction, attention analysis, cognitive processing, detecting human fatigue and micro-emotions. Many researchers are working on improving the existing eye tracking methodologies by collecting large amounts of eye images and using it to train predictive models that can track the eye and its movements. Modern video-based eye-trackers use infrared cameras to monitor movements of the eyes in order to gather information about the visual behavior and perceptual strategies of individuals engaged in various tasks. Eye-trackers have traditionally been mounted to computer screens or worn directly on the head for mobile applications and are increasingly being embedded in head-mounted-displays to support rendering and interaction in virtual and augmented reality applications. Contemporary algorithms for estimating gaze direction rely heavily on the segmentation of specific regions of interest in the eye images, such as the pupil, iris, sclera (see Figure 1.1). These features are then used to estimate the causal 3D geometry, in the form of a 3D model of the spherical eye in camera space [23]. Segmentation is complicated by the presence of corrective lenses or by reflections of the surrounding environment upon intervening physiology, such as the cornea and the tear layer. Convolutional neural networks (CNNs) have presented a promising new approach for the annotation of eye images even in these challenging conditions [8, 28, 27, 11, 13, 24]. However promising, the use of CNNs trained through supervised learning requires a large corpus of annotated eye images. Manual annotation is time-intensive and, although these datasets exist, they are susceptible to errors introduced during human annotation, and only a

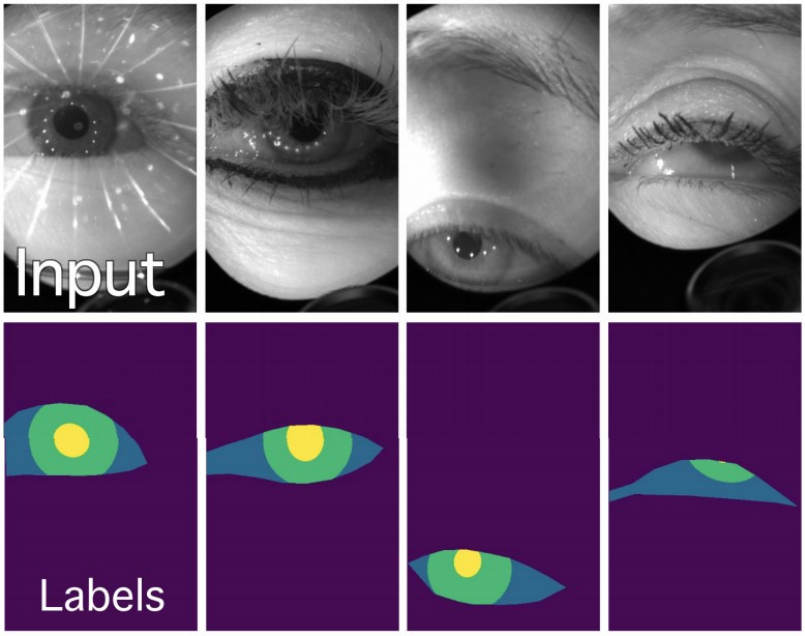

Figure 1.1: Real eye images and corresponding segmented eye region labels. Image courtesy of RITNet [4].

few exist that include manually segmented image features other than the pupil center which are necessary for building an accurate 3D model of the eye. For a comprehensive list of existing datasets, please refer to Garbin *et al.* [9]. Although existing dataset images may include reflections of the surrounding environment, their inclusion has been unsystematic, and their contributions to the robustness of segmentation remain unclear.

To alleviate the cumbersome process of manual labeling, computer graphics rendering is employed to automatically generate a large corpus of annotated eye images under various conditions. The images and ground truth generated using computer graphic will be pixel perfect because labelling done is done as preprocess before the images are generated by using the appropriate texture for each class object. Real-world eye image capture is time consuming and is a very tedious process which can be avoided by rendering synthetic dataset which is fast easy and cheap. And there is better control over the size and resolution of the rendered images. Uniform distribution of the dependent parameters like iris

,skin textures, gaze position, camera position and illumination, etc. can be achieved which is very important for neural network training. In this work, we introduce a synthetic eye image and video generation platform called RIT-Eyes that improves upon previous work by adding features such as an active deformable iris, an aspherical cornea, retinal retro-reflection, gaze-coordinated eye-lid deformations, and blinks. To demonstrate the utility of our platform, we render images reflecting the represented gaze distributions inherent in two publicly available eye image datasets, NVGaze and OpenEDS. Our platform enables users to render realistic eye images by providing parameters for camera position, illuminator position, and head and eye pose. The pipeline can also be used to render temporal sequences of realistic eye movements captured in datasets such as Gaze-in-Wild.

Chapter 2

Related Works

The use of synthetic eye images has been proposed by other researchers including Swirski *et al.*[22], Wood *et al.*[26, 25], and Kim *et al.*[11]. Their work involves rendering near-eye images in which the location of image features is known, circumventing the need for manual annotation. Example renderings from each of these works shown in figure 2.1.

Figure 2.1: Rendered images from Swirski *et al.*[22] (left), Wood *et al.*[26, 25] (middle), and Kim *et al.*[11] (right).

Our synthetic eye image generation platform introduces several improvements including an accurate aspherical corneal model, a deformable iris, the lacrimal caruncle (the small pink nodule located at the inner corner of the eye), deformable eyelids, and a retina with retro-reflective properties which can aid in developing 'bright pupil' solutions [14] for head-mounted or remote eye trackers.

The real evaluation of the effectiveness of any synthetic dataset lies in its ability to be leveraged in real-world problems. Although initial efforts demonstrate that CNNs trained on artificial stimuli for semantic segmentation can generalize to true imagery [11, 20, 19],

these tests are limited to specific applications. Kim *et al.*[11] showed that despite their best efforts to model realistic distributions of natural eye imagery in a virtual reality headset, training on synthetic eye images while testing on real data resulted in a 3.1° accuracy error on average, which is $\sim 1^\circ$ higher than training on real imagery. Park *et al.*[18] utilized the UnityEyes dataset [25] to train a stacked hourglass architecture [16] to detect landmarks in real-world eye imagery. By augmenting the synthetic data with a few real-world images, they observed an improvement in performance. Together these studies suggest that the underlying distribution within existing synthetic datasets cannot capture the variability observed in real world eye imagery. While techniques such as few shot learning [18] and joint learning/un-learning [2] may help combat these issues, an inherently better training set distribution should improve the performance of CNNs.

To support the development of CNNs for semantic segmentation, we used our novel synthetic eye image generation platform to render three synthetic datasets: two that approximate the eye/camera/emitter position properties of publicly available datasets, one synthetic - NVGaze [11], and one of real eye imagery - OpenEDS [9], and a third dataset that approximates the eye/camera/emitter properties of the Pupil Labs Core wearable eye tracker [10], referred to as S-General. Renderings which mimic NVGaze-synthetic and OpenEDS will be referred to as S-NVGaze and S-OpenEDS respectively. These datasets enable us to test the generalizability of our rendering pipeline. For example, if two CNNs trained on S-NVGaze and S-OpenEDS respectively exhibit little or no difference in performance when tested on an external image, we can conclude that properties differentiating S-NVgaze and S-OpenEDS (namely the camera orientation, and placement and number of emitters) do not contribute to the semantic understanding of different eye regions in the external image. A fourth dataset called S-Natural which stimulates images captured in outdoor environments has also been rendered and is available for research purposes.

Chapter 3

Implementation

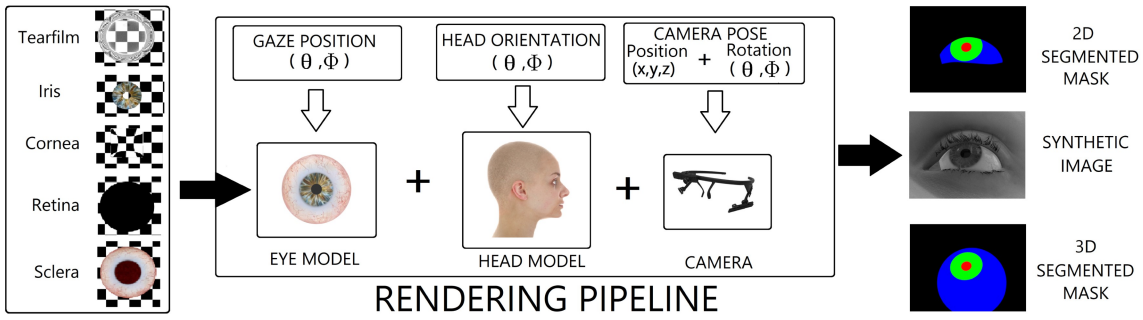

Figure 3.1: RIT-Eyes Rendering Pipeline.The components of the eye model are shown on the left. The user selects a head model and the pipeline takes as additional input gaze position, head orientation, camera pose. The output of the pipeline includes a synthetic image along with corresponding 2D and 3D segmented masks.

The following section explains the rendering pipeline of RIT-Eyes. The entire pipeline is rigged and modelled in Blender 2.8 and is available for researchers for eye tracking and deep learning applications. RIT-Eyes allows users to render images and videos with the constraints provided by the users with no blender experience. Figure 3.1 illustrates the rendering pipeline. Table 3.1 summarizes the features offered by our rendering pipeline compared to previous work.

Properties	Unity Eyes	SynthesEyes	NVGaze	Swirski	Ours
Aspherical cornea	×	×	×	×	✓
Retroreflection	×	×	×	×	✓
Segmentation mask	×	×	✓	×	✓
Infrared rendering	×	×	✓	✓	✓
RGB rendering	✓	✓	×	×	✓
Reflective eye-wear	×	×	✓	×	✓
Lacrimal caruncle	×	×	×	×	✓
Variable eyelid position	×	×	✓	✓	✓
Real-time rendering	✓	×	×	×	×
Sequential Rendering	×	×	×	×	✓

Table 3.1: Comparison between RIT-Eyes and existing synthetic rendering pipelines.

3.1 Head Models

Our rendering platform currently incorporates 24 head models (12 male, 12 female) with varying skin color, gender, and eye shape (see Figure 3.2). The head models were purchased from an online repository[1]. The associated textures contain 8K color maps captured using high-quality cameras. To approximate the properties of human skin in the infrared domain, the red channel from the original diffuse texture map is incorporated for rendering purposes.

To overcome the challenge of controlling the placement of eyelashes relative to the eye-lid position, we replaced each of the model's original eyelashes with Blender's built-in hair particle editor which provides a plausible physical simulation of hair behavior. This is similar to the approach used by Swirski et al. [22]. We also replaced the basic 3D eyeball included with the 3D head models with our own customized 3D eyeball that provides greater control and more faithfully simulates the structure of real eyes. Figure 3.3 shows the modelling view of the head, eye and camera arrangement in Blender.

[1] `https://www.3dscanstore.com/`

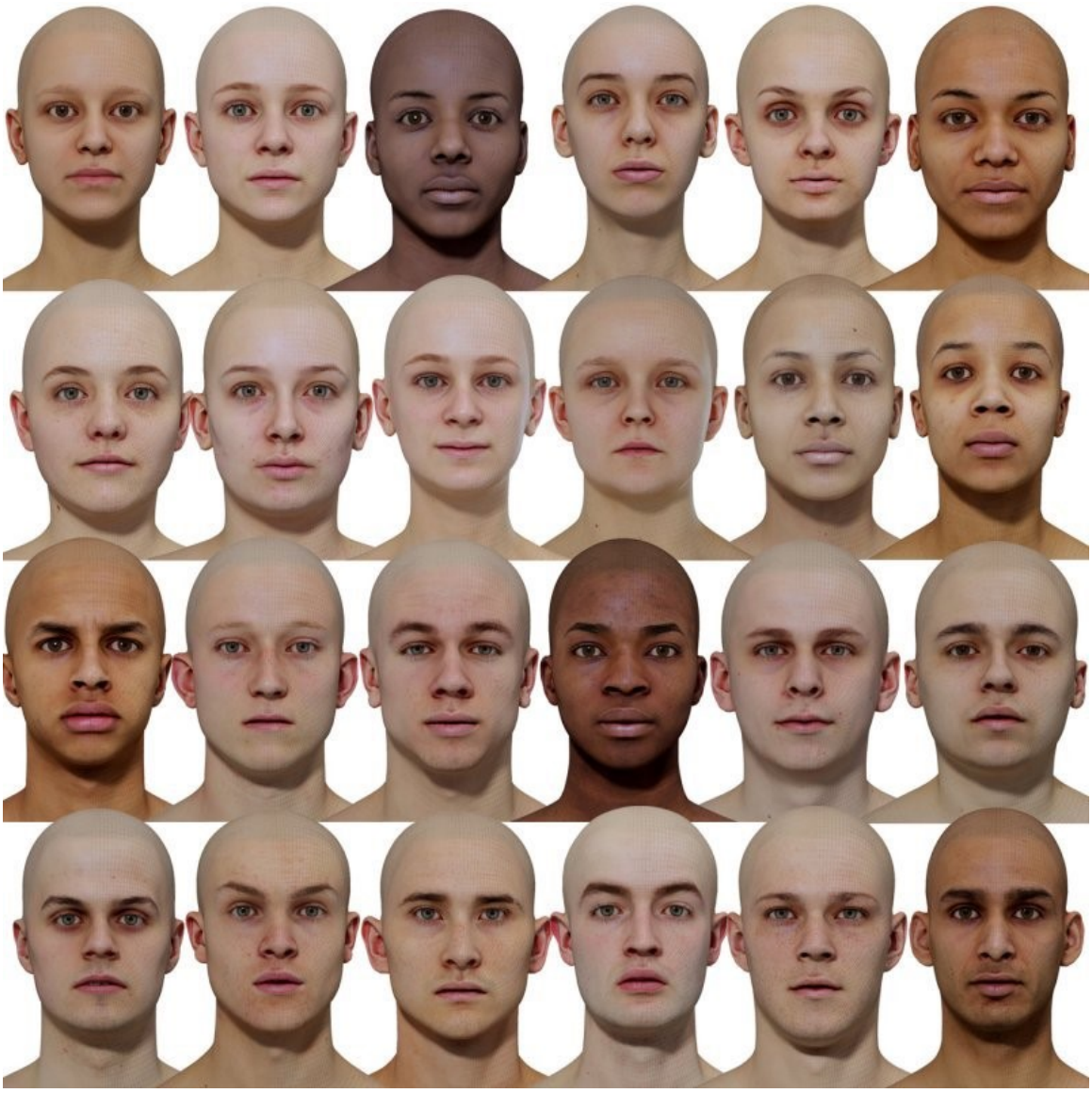

Figure 3.2: Head models used in RIT-Eyes. Image courtesy of `https://www.3dscanstore.com/`

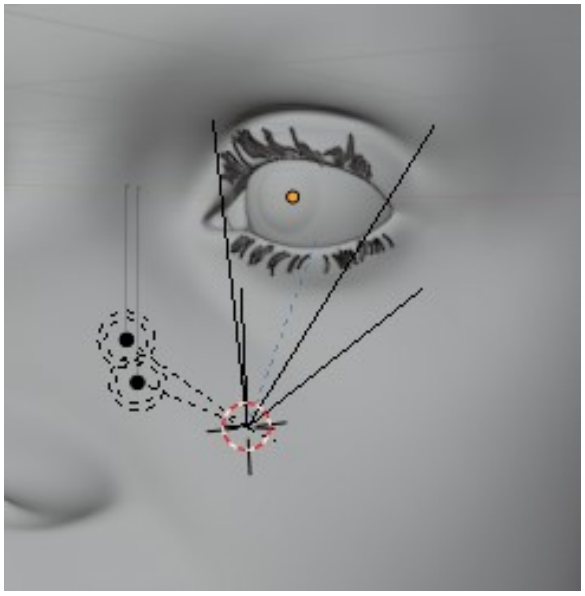

Figure 3.3: Blender view of the head, eye and camera orientation.

3.2 Eye Model

Reconstructing all parameters that influence the imaging of a person's eye is a difficult task, and it is common to make simplifying assumptions regarding its structure. We used a modified Emsley-reduced optical model of the human eye [3, 6]. Table 3.2 summarizes the various basic physical properties. Modeling and rendering were accomplished using Blender-2.8.

Feature	**Radius (mm)**	**Refractive index (n)**
Cornea	7.8 mm	1.3375
Pupil	1-4 mm	×
Iris disc	6 mm	×
Eyeball sphere	12 mm	×

Table 3.2: Basic physical properties of our eye model.Radius and refractive index values courtesy of Dierkes et al. [6]

Furthermore, our eye model incorporates the following features:

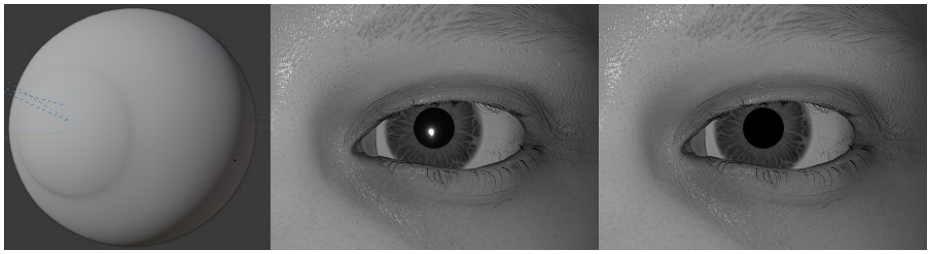

Figure 3.4: Comparison renderings to illustrate improvements offered by our model. With tear film (middle). Without (right). Blender view of tear film (left).

Figure 3.5: Comparison renderings to illustrate improvements offered by our model. With aspherical cornea (middle). With no cornea (right). Blender view of aspherical cornea (left).

Tear film: Similar to previous work [26], we designed a tear film on the outermost surface of the eyeball with glossy and transparent properties to produce plausible environmental reflections on the surface of the eye (see Figure 3.4).

Aspherical cornea: In contrast to previous work, we chose to render a physiologically accurate corneal bulge (see Figure 3.5). The corneal topography is modeled as a spheroid, $x^2 + y^2 + (1 + Q)z^2 - 2Rz = 0$ [7], where Q is the asphericity and R is the corneal radius of curvature. Research has shown that the human eye exhibits Q value of $\mu = -0.250$, $\sigma = 0.12$ [7]. Our corneal models incorporate three asphericity values, -0.130, -0.250 and -0.370, which were represented uniformly during rendering.

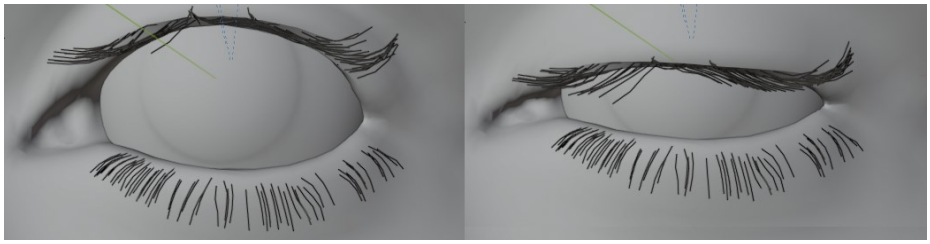

Figure 3.6: Blender view of the eye lid deformation.

Deformable eyelids: In order to avoid any visible gaps between the eyelids and the eyeball, the original 3D vertices in each eye socket were morphed to a snug fit around our custom eyeball (see Figure 3.6). Using Blender's inbuilt wrapping function, we deformed the eyelid mesh to conform to the corneal contour below it. To mimic human behavior, the amount of eyelid closure was approximated by a linear function of eye rotation in the vertical axis (see Figure 3.7).

Pupil aperture: Previous datasets have modeled the pupil as an opaque black disc. The pupil in our eye model was accurately modeled as an aperture (see Figure 3.9) such that constriction or dilation of the pupil was accompanied by appropriate deformation of surrounding iris texture (see Figure 3.8). The pupil aperture opening was uniformly distributed between 1 mm to 4 mm in radius [6].

Lacrimal caruncle: In contrast to previous works, we included the lacrimal caruncle, a small pink nodule positioned at the inner corner of the eye (see Figure 3.10). The lacrimal caruncle consists of skin, sebaceous glands, sweat glands, and hair follicles. Hence when generating the segmentation mask of the respective synthetic eye image, the lacrimal caruncle was considered to be part of the skin (see Figure 3.10) as opposed to OpenEDS [9], which segmented lacrimal caruncle as a part of sclera.

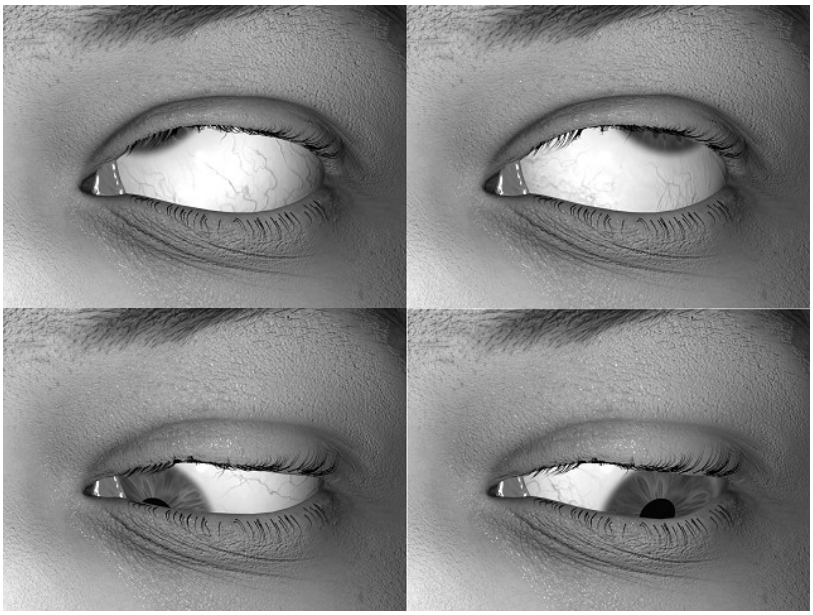

Figure 3.7: Renderings to illustrate improvements offered by our model. Eye lid deformation (shown at extreme gaze angles).

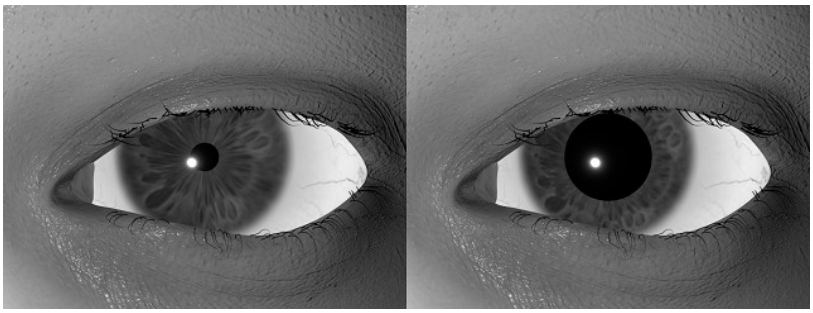

Figure 3.8: Renderings to illustrate improvements offered by our model. Variable size pupil aperture. 1 mm radius (left) to 3 mm radius (right).

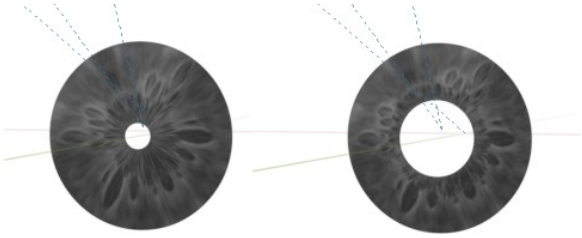

Figure 3.9: Blender view of the pupil size variation.

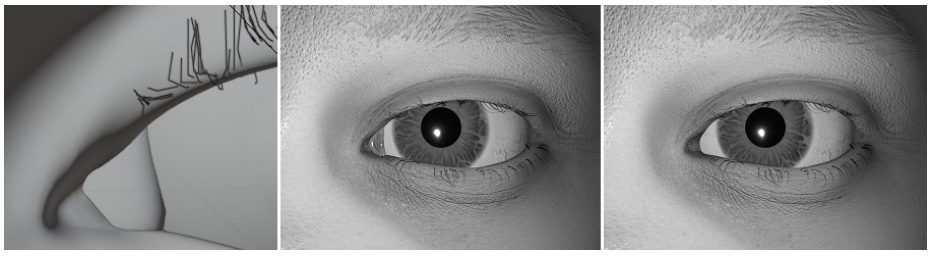

Figure 3.10: Comparison renderings to illustrate improvements offered by our model. With lacrimal caruncle (middle). Without (right). Blender view of lacrimal caruncle (left).

Bright pupil response: The bright pupil response occurs when a light source is within $\sim$2.25° of separation from the imaging optical axis [17]. To simulate eye physiology, we added retroreflectivity to the retinal wall. Reflectivity increases as the angle of separation decreases following a Beckmann distribution (see Figure 3.11).

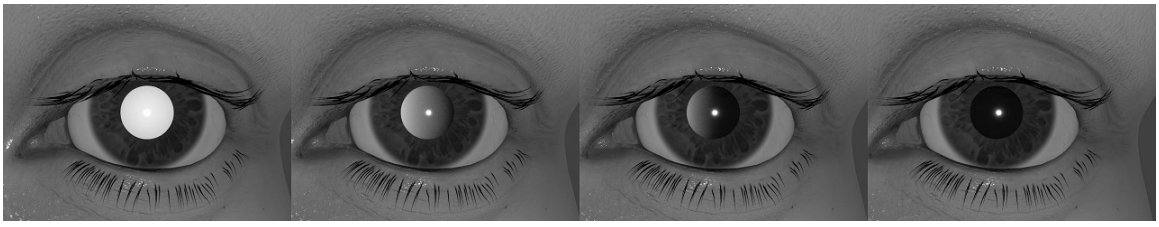

Figure 3.11: Renderings to illustrate improvements offered by our model. Bright pupil effect at varying degrees of angular separation between the imaging optical axis and the light source. From left to right: 0°, 1.16°, 1.51°, 2.25°.

Environment mapping and reflective eye-wear: Following Wood et al. [26], we used 360° HDR images to simulate the reflections from the external environment. The environment texture is mapped onto a sphere with the eye model at its center. Each pixel intensity on the texture acts as a separate light source that illuminates the model. We used 25 HDR images obtained from an online repository [2] (see Figure 3.13). Of these, nine were indoor,

[2] `https://hdrihaven.com/`

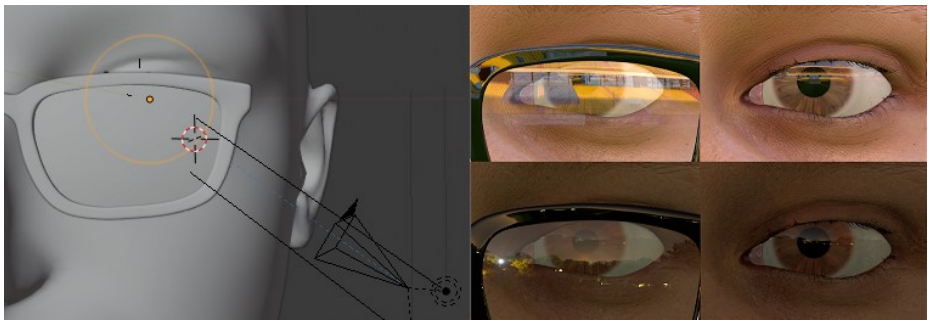

Figure 3.12: Renderings showing environmental mappings with (middle) and without (right) glasses. Blender view of the model with glasses (left).

and 16 were outdoor scenes (see Supplementary Material). The pixel intensity of the environment texture was varied between $\pm 50\%$ of its original value. Textures were chosen at random, and their global intensity scaled to ensure equal proportions of dark, well lit, and saturated imagery. The environment map was rotated up to 360° along the y-axis and up to $\pm 60^\circ$ on the x and z-axis to induce a unique reflection pattern on the model for every rendered image. Figure 3.12 shows examples of environment mapping with and without reflective eye-wear.

Iris and sclera textures Our rendering platform currently incorporates 9 infrared (IR) textures of the iris 7 obtained using IR photography of the human eye (courtesy of John Daugman [5]) and 2 artificial renderings previously used by Swirksi et al. [21]. Note that among the 7 photographed textures of the iris, parts may be occluded due to eye lashes, eye lid position, or by reflections. In order to remove these artifacts, the images were manually edited using Photoshop. The texture for the sclera was purchased from an online repository [3]. Since, we only had access to one sclera texture and 9 iris textures, a random rotation between 0° and 360° was applied to the sclera and iris textures to increase variability in the rendered eye images.

[3] `https://www.cgtrader.com/`

Figure 3.13: The 25 HDR environment maps used during rendering to simulate realistic lighting.

Chapter 4

Rendered Dataset

In addition, eye pose will be uniformly distributed within $\pm 30^{\circ}$ in both azimuth and elevation. For each rendered image in the dataset, we will generate ground truth masks of the sclera, iris, and pupil with and without the skin (see Figure: 4.1). We will also record additional metadata, including the 2D and 3D center of various eye features relative to the camera, as well as eye pose in degrees, and the camera intrinsic matrix.

4.0.1 S-OpenEDS

OpenEDS [9] is a large dataset that contains real eye images that were captured using an HMD at 200 Hz. Among the images present in the dataset, our work mainly focuses on the 12,759 real eye images which had annotated segmented masks. Replicating the image properties and eye position, we will render images of 400 x 640 pixels at 200 rays-per-pixel (see Figure 4.3). Note that, in order to approximate the lighting conditions of the OpenEDS dataset, 16 point-source light sources were arranged around the virtual eye camera in a

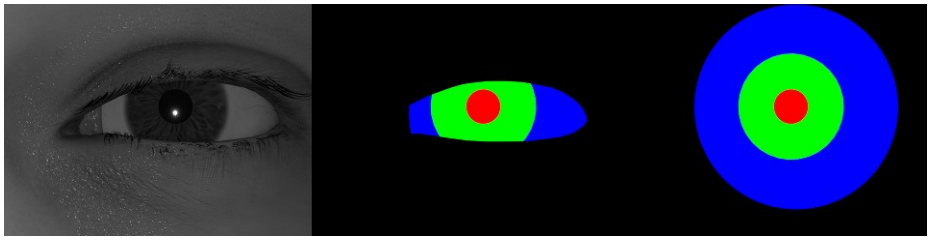

Figure 4.1: Sample image along with groundtruth mask of pupil (red), iris (green), and sclera (blue) with and without skin.

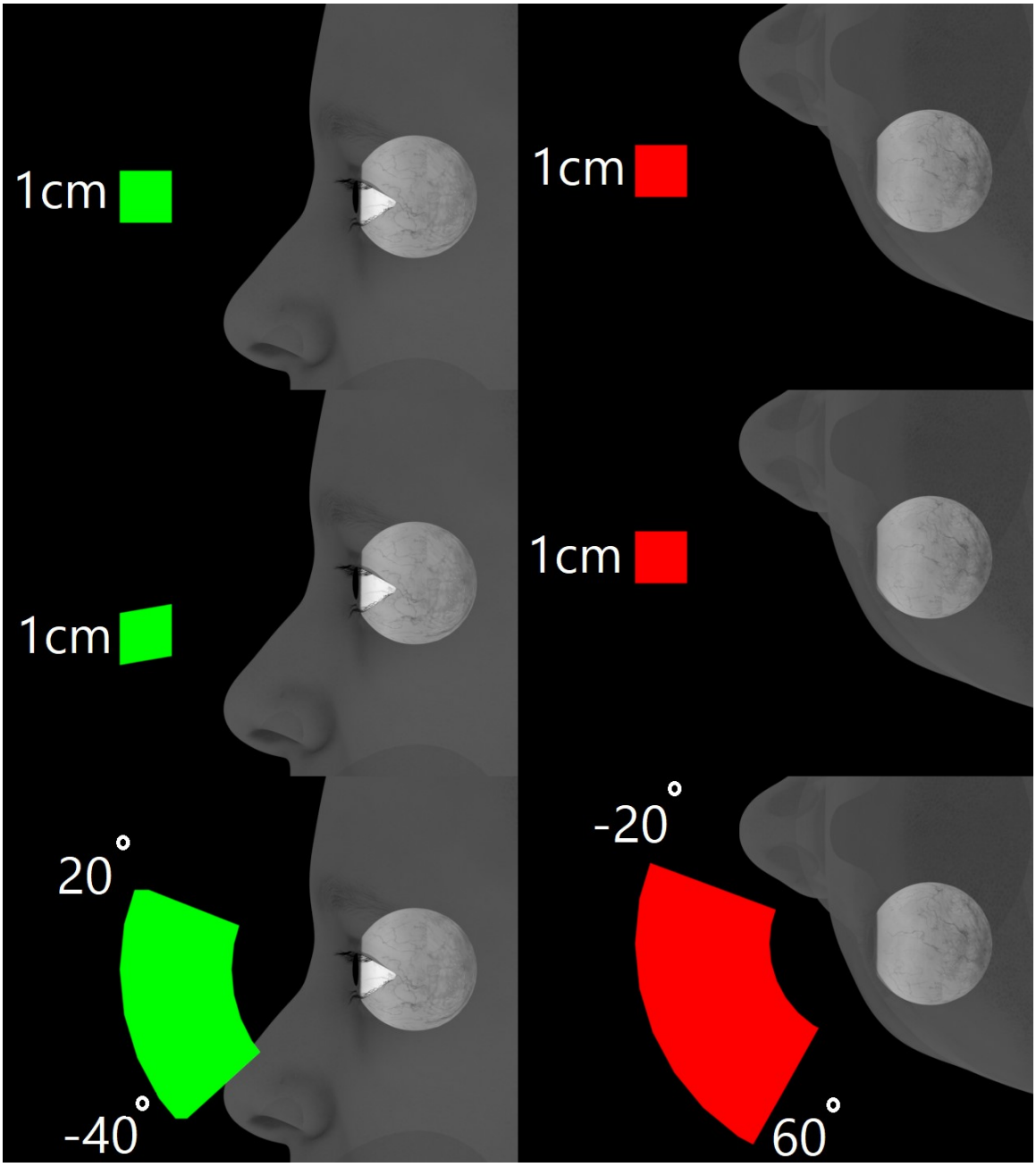

Figure 4.2: Camera positions used for S-NVGaze (top), S-OpenEDS (middle), and S-General (bottom). Side-view (left column). Top-view (right column).

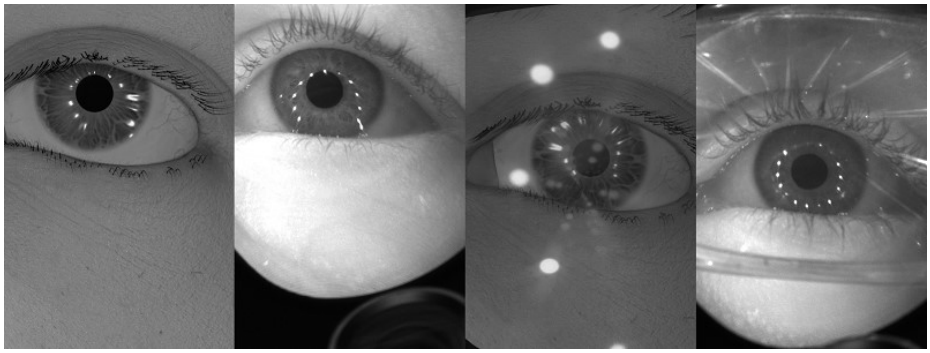

Figure 4.3: Comparison of images from S-OpenEDS (odd columns) with corresponding images from OpenEDS (even columns).

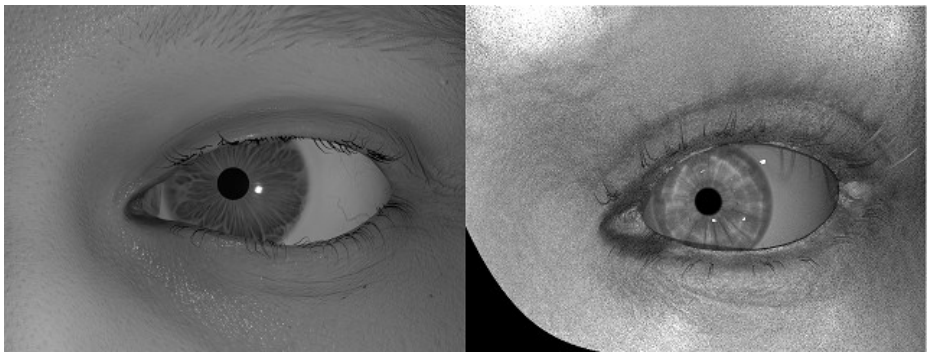

Figure 4.4: Comparison of image from S-NVGaze (left) with corresponding image from NVGaze (right).

circular pattern. Blender's native compositor was used to imitate the resulting pattern of reflection (referred to as starburst in [4]) upon the iris from the light sources. The camera position was uniformly shifted ± 5 mm in the horizontal axis and vertical axis in-order to account for the slippage of the head-mounted camera. The camera was rotated +10 along the x-axis and the distance was uniformly varied from 3.5 cm to 4.5 cm from the tip of the eye based on empirical observations.

4.0.2 S-NVGaze

The NVGaze dataset contains 2M images of synthetically generated eye images taken using an on-axis camera configuration. The S-NVGaze images were rendered at a resolution of 640 x 480 pixels with 200 rays per pixel (see Figure 4.4). Similar to S-OpenEDS, the camera will be placed in front of the eyes within an emperically derived distance of 3.5 cm to 4.5 cm from the tip of the eye. Eye camera position will be varied $\pm$5 mm vertical and horizontal axis to simulate the headgear slippage conditions. Unlike S-OpenEDS, one point light sources will be placed 2 mm to the anterior and nasal side of the eye camera.

4.0.3 S-General

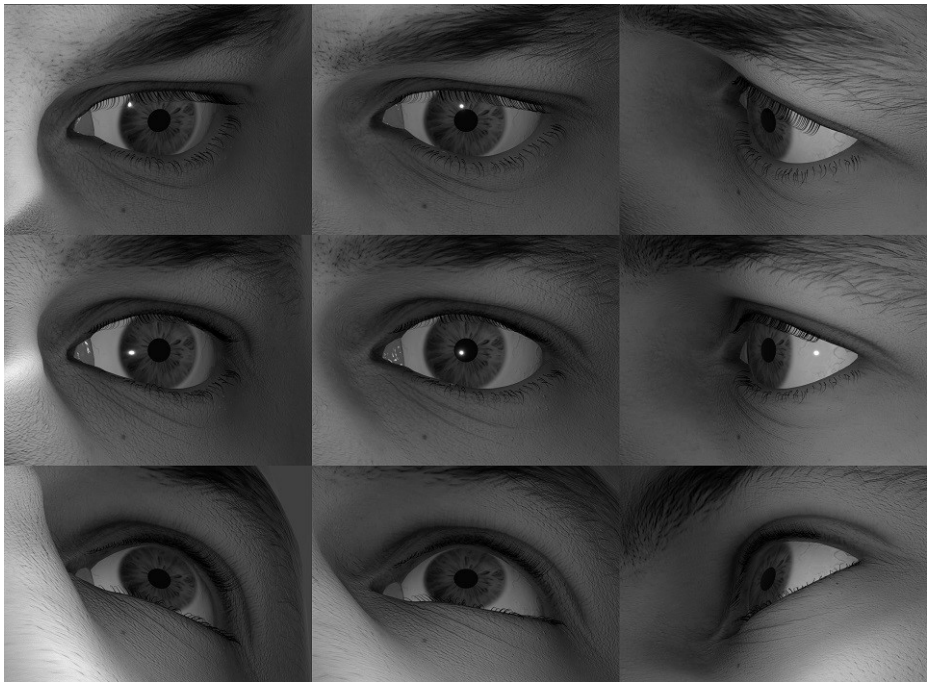

Figure 4.5: Sample images of eye with fixed eye gaze and varying camera position. The camera position at -20°, 0°, 60° (left to right) in azimuthal plane and -20°, 0°, 40° in elevation plane (top to bottom).

This dataset approximates the conditions imposed by the Pupil Labs eye mobile tracker.

The camera position will be uniformly distributed within an eye-centered spherical manifold subtending -20° to 60° azimuthal and -20° to 40 ° in elevation (see Figure 4.2), which supersets the range of camera positions afforded by the Pupil Labs mobile eye tracker (see Figure 4.5). Note that a smaller jitter of ± 1 mm in the vertical and horizontal plane will also be added to account for possible variation in the eye tracker's slippage. The major difference to S-NVGaze and S-OpenEDS is a wider variation in camera position and orientation. Additionally, only one point-source light source is used as in original S-NVGaze to replicate most of the available commercial eye-trackers.

4.0.4 S-Natural

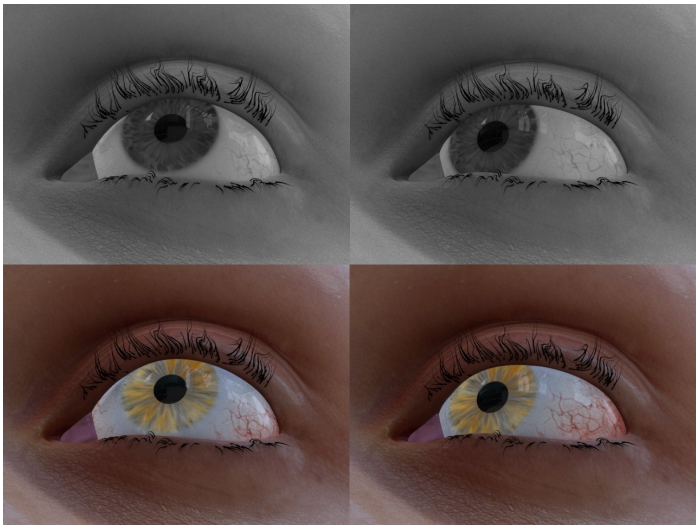

Figure 4.6: Simulated infrared rendering (top) vs RGB rendering (bottom)

While we do not render any reflections in S-OpenEDS, S-NVgaze or S-General, we publish an auxiliary dataset, S-Natural (see Figure 4.6) which shares the same distributions and annotations as S-general while rendered in color with ambient reflections. Note that we do not conduct any experiments on this dataset and is provided for researchers interested in developing gaze mapping solutions under naturalistic conditions. RGB renderings incorporated ten artificial renderings of the iris illuminated within the visible-range [26].

4.0.5 Sequential Renderings

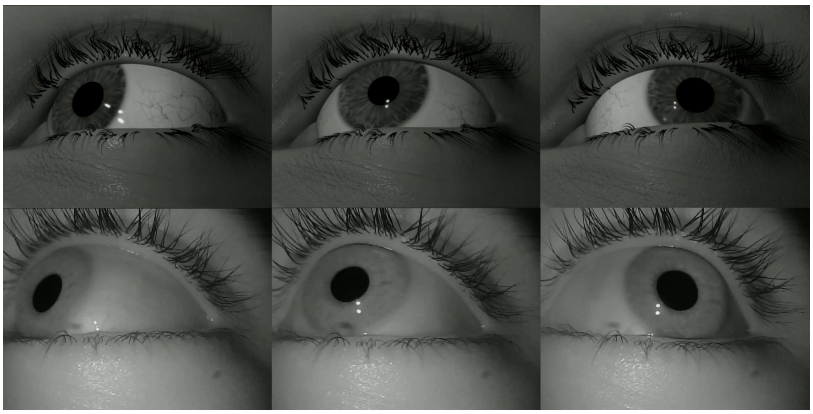

Figure 4.7: Sequential renderings (Top) using Gaze-in-Wild data vs Corresponding

Datasets such as Gaze-in-Wild[12] and 360em[1] provide labeled sequences of eye and head movements. Our current rendering pipeline allows us to leverage the eye and head pose information from these datasets to render artificial sequences with similar pose, and environmental effects, occlusions, naturalistic blinks and pupil size variation (see Figure 4.7).

Chapter 5

Applications

Four new datasets were rendered for use in advancing eye tracking research. The new datasets *S-NVGaze* and *S-OpenEDS* are synthetic renderings intended to mimic the NVGaze [11] and OpenEDS [9] datasets, respectively. The third and fourth dataset, *S-General* and S-Natural reflects the camera parameters of the pupil Labs mobile eye tracker. Each new dataset includes 48,000 path-traced images rendered using Blender's Cycles rendering engine for a total of 192,000 images. Possible applications of the rendered datasets include:

Semantic Segmentation Eye segmentation is a common method used in eye tracking applications for improving the performance of eye-gaze estimation and pupil center detection. The datasets have already been used to train CNN models to segment the input images into 4 classes (pupil, iris, sclera, and background) (see Figure 5.1) and to test the ability for these CNNs to generalize across datasets collected using different configurations of the eye, camera, and infrared emitter(s) [15].

Style Transfer Advances in generative adversarial networks(GANs) have shown great promise for improving style transfer from one image to another. GANs have been used to refine the appearance of images from the Unity Eyes synthetic dataset [22]. The improved appearance resulted in a smaller gaze error when compared to the unrefined images. Alternatively, it has been observed that the ability to generalize to real world imagery improves when a small number of hand-labelled real world eye images is included into the training set of synthetic eye imagery [13]. We plan to extend our rendering pipeline to leverage

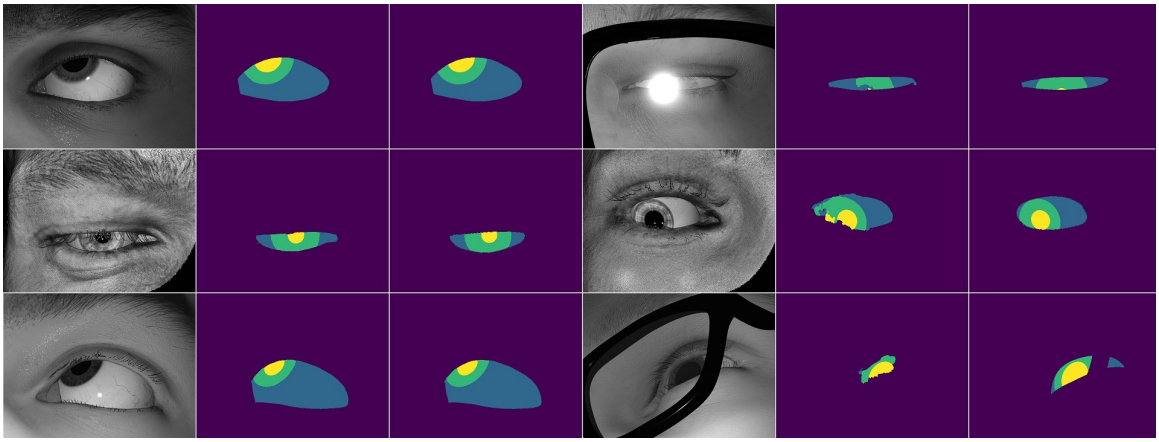

Figure 5.1: Semantic Segmentation results

similar approaches. Figure 5.2 shows the results of applying style transfer to one image from our dataset. One promising use of style transfer is for privacy preservation in AR/VR applications while still allowing the eye images to be functional for eye-tracking.

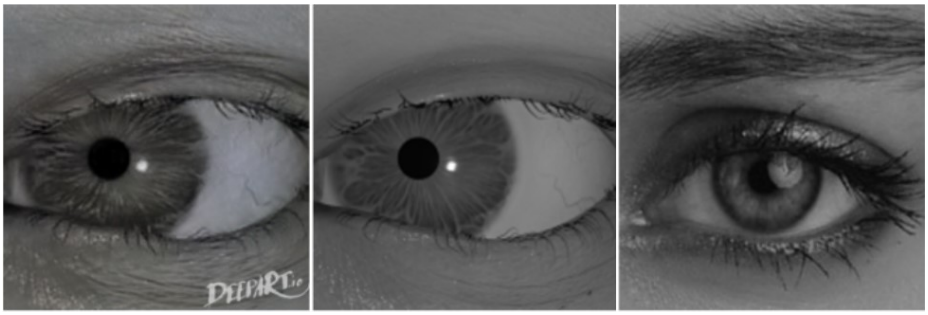

Figure 5.2: Style transfer results. The textures from style image (right) and eye pose from reference image (middle) results in the generated image (left)

Chapter 6

Conclusions

6.1 Limitations and Future Work

Although our work provides improvements to various eye features, it is still based on a simplified eye model and thus there is room for improvement. Several assumptions we made in an attempt to replicate the visual effects for example there are no corneal reflections but similar reflections were achieved by adding the tear film around. Moreover retro reflection effect was achieved by adjusting the intensity of the reflection and not by making an actual retro reflective object. Better control can also be obtained by rigging the entire model using bones (see Figure 6.1) instead of wrapping functions but the drawback is that it will make the rendering pipeline significantly slower. We plan to explore if incorporating such information in our rendering pipeline can enhance the visual appearance of the synthetic eye imagery and improve performance of gaze estimation. Although our datasets include near-eye images with eyeglasses, the simulated glasses currently only reflect incoming light, they did not refract light. Similarly, the presence of makeup, such as eye liner, eye shadow or mascara, which have been shown to interfere with many conventional algorithms for pupil detection and gaze estimation [9, 11], is not accounted for. Earlier works also do not address this. At 640x 480px with 200 rays per pixel, rendering the synthetic image and the corresponding ground truth mask takes around 8 seconds. This is a major drawback as compared to Unity Eyes [26], which renders images in real time. The way to increase the rendering speed would be to move the pipeline to unity.

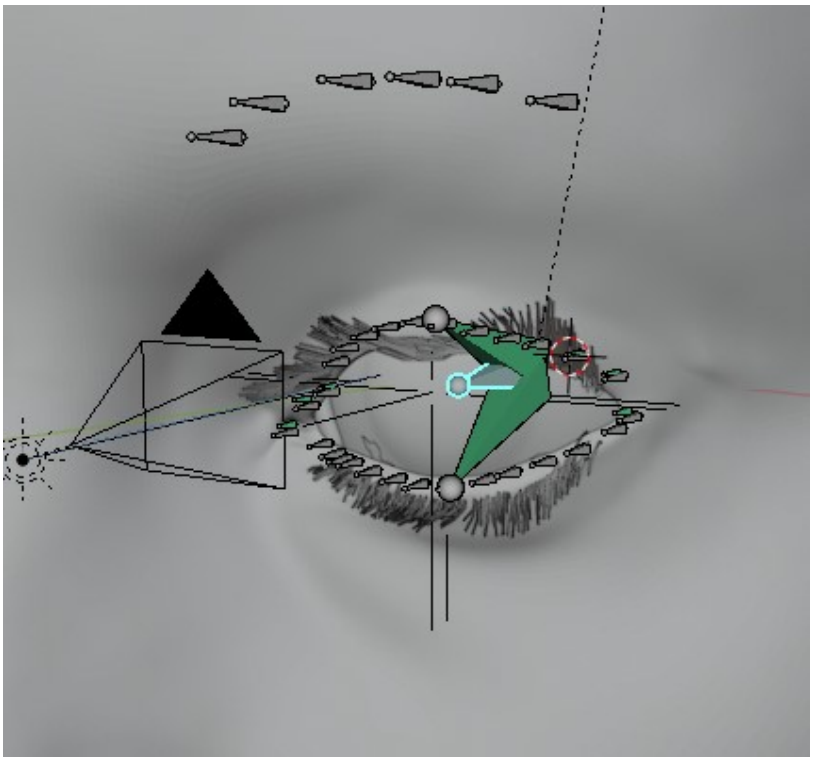

Figure 6.1: Eyelid rigged using weighted bones.

6.2 Conclusion

This thesis presents a novel synthetic eye image generation platform that provides several improvements over existing work to support the development and evaluation of eye-tracking algorithms. This platform is used to render synthetic datasets, S-NVGaze and S-OpenEDS, reflecting the spatial arrangement of eye cameras in two publicly available datasets, NVGaze and OpenEDS. We also render two datasets which mimic the characteristics of Pupil Labs Core mobile eye tracker. The rendered datasets along with converged models for semantic segmentation are made publicly available to aid researchers in developing novel gaze tracking solutions.

Bibliography

[1] Ioannis Agtzidis, Mikhail Startsev, and Michael Dorr. A Ground-Truth Data Set and a Classification Algorithm for Eye Movements in 360-degree Videos. 2019.

[2] Mohsan Alvi, Andrew Zisserman, and Christoffer Nellåker. Turning a blind eye: Explicit removal of biases and variation from deep neural network embeddings. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11129 LNCS:556–572, 2019.

[3] David A. Atchison and Larry N. Thibos. Optical models of the human eye. *Clinical and Experimental Optometry*, 99(2):99–106, 2016.

[4] Aayush K. Chaudhary, Rakshit Kothari, Manoj Acharya, Shusil Dangi, Nitinraj Nair, Reynold Bailey, Christopher Kanan, Gabriel Diaz, and Jeff B. Pelz. RITnet: Real-time Semantic Segmentation of the Eye for Gaze Tracking. pages 1–5, 2019.

[5] John Daugman. How Iris Recognition Works. *The Essential Guide to Image Processing*, 14(1):715–739, 2009.

[6] Kai Dierkes, Moritz Kassner, and Andreas Bulling. A novel approach to single camera, glint-free 3D eye model fitting including corneal refraction. *Eye Tracking Research and Applications Symposium (ETRA)*, (June), 2018.

[7] Georges M. Durr, Edouard Auvinet, Jeb Ong, Jean Meunier, and Isabelle Brunette. Corneal Shape, Volume, and Interocular Symmetry: Parameters to Optimize the Design of Biosynthetic Corneal Substitutes. *Investigative Opthalmology & Visual Science*, 56(8):4275, 7 2015.

[8] Wolfgang Fuhl, Thiago Santini, Gjergji Kasneci, Wolfgang Rosenstiel, and Enkelejda Kasneci. PupilNet v2.0: Convolutional Neural Networks for CPU based real time Robust Pupil Detection. 10 2017.

[9] Stephan J. Garbin, Yiru Shen, Immo Schuetz, Robert Cavin, Gregory Hughes, and Sachin S. Talathi. OpenEDS: Open Eye Dataset. 4 2019.

[10] Moritz Kassner, William Patera, and Andreas Bulling. Pupil: An open source platform for pervasive eye tracking and mobile gaze-based interaction. *UbiComp 2014 - Adjunct Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 1151–1160, 2014.

[11] Joohwan Kim, Michael Stengel, Alexander Majercik, Shalini De Mello, David Dunn, Samuli Laine, Morgan McGuire, and David Luebke. NVGaze: An anatomically-informed dataset for low-latency, near-eye gaze estimation. *Conference on Human Factors in Computing Systems - Proceedings*, 12:1–12, 2019.

[12] Rakshit Kothari, Zhizhuo Yang, Christopher Kanan, Reynold Bailey, Jeff B. Pelz, and Gabriel J. Diaz. Gaze-in-wild: A dataset for studying eye and head coordination in everyday activities. *Scientific Reports*, pages 1–23, 2020.

[13] Erik Lindén, Jonas Sjöstrand, and Alexandre Proutiere. Learning to Personalize in Appearance-Based Gaze Tracking. 7 2018.

[14] Carlos H. Morimoto and Marcio R.M. Mimica. Eye gaze tracking techniques for interactive applications. *Computer Vision and Image Understanding*, 98(1):4–24, 2005.

[15] Nitinraj Nair, Rakshit Kothari, Aayush K Chaudhary, Zhizhuo Yang, Gabriel J Diaz, Jeff B Pelz, and Reynold J Bailey. Rit-eyes: Rendering of near-eye images for eye-tracking applications. *arXiv preprint arXiv:2006.03642*, 2020.

[16] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2016.

[17] Karlene Nguyen, Cindy Wagner, David Koons, and Myron Flickner. Differences in the infrared bright pupil response of human eyes. *Eye Tracking Research and Applications Symposium (ETRA)*, pages 133–138, 2002.

[18] Seonwook Park, Shalini De Mello, Pavlo Molchanov, Umar Iqbal, Otmar Hilliges, and Jan Kautz. Few-shot Adaptive Gaze Estimation. 5 2019.

[19] Seonwook Park, Adrian Spurr, and Otmar Hilliges. Deep Pictorial Gaze Estimation. volume 11217 LNCS, pages 741–757. 2018.

[20] Seonwook Park, Xucong Zhang, Andreas Bulling, and Otmar Hilliges. Learning to find eye region landmarks for remote gaze estimation in unconstrained settings. In *Eye Tracking Research and Applications Symposium (ETRA)*, pages 1–10, New York, New York, USA, 2018. ACM Press.

[21] Lech Świrski. Gaze estimation on glasses-based stereoscopic displays. (August), 2015.

[22] Lech Świrski and Neil Dodgson. Rendering synthetic ground truth images for eye tracker evaluation. In *Eye Tracking Research and Applications Symposium (ETRA)*, pages 219–222, 2014.

[23] Lech Świrski and Neil A. Dodgson. A fully-automatic, temporal approach to single camera, glint-free 3D eye model fitting. *Pervasive Eye Tracking and Mobile Eye-Based Interaction (PETMEI)*, 2013.

[24] F. J. Vera-Olmos, E. Pardo, H. Melero, and N. Malpica. DeepEye: Deep convolutional network for pupil detection in real environments. *Integrated Computer-Aided Engineering*, 26(1):85–95, 2018.

[25] Erroll Wood, Tadas Baltruaitis, Xucong Zhang, Yusuke Sugano, Peter Robinson, and Andreas Bulling. Rendering of eyes for eye-shape registration and gaze estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 2015 Inter, pages 3756–3764, New York, NY, 12 2015. IEEE.

[26] Erroll Wood, Tadas Baltrušaitis, Louis Philippe Morency, Peter Robinson, and Andreas Bulling. Learning an appearance-based gaze estimator from one million synthesised images. *Eye Tracking Research and Applications Symposium (ETRA)*, 14:131–138, 2016.

[27] Yuk Hoi Yiu, Moustafa Aboulatta, Theresa Raiser, Leoni Ophey, Virginia L. Flanagin, Peter zu Eulenburg, and Seyed Ahmad Ahmadi. DeepVOG: Open-source pupil segmentation and gaze estimation in neuroscience using deep learning. *Journal of Neuroscience Methods*, 324:108307, 2019.

[28] Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling. Appearance-based gaze estimation in the wild. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 07-12-June:4511–4520, 2015.